

Basic Operation

The next section will discuss some of the software organisation dealing less with hardware driving and more with data processing and flow control.

The scales effectively perform the same function for the majority of the time. – Counting by weight is effectively weighing in custom units. Displaying raw readings in factory mode is also weighing, only without converting to predefined units.

With the exception of factory mode only functions, the entire functionality can be summed up in one flow chart.

The processes shown are passed values corresponding to current operating unit and user mode. – Hence the divide by units value and display weight will behave differently depending on what mode the scales are in.

Any update of the settings is backed up in EEPROM on update. – It is important that this happens when the settings so that an unexpected power-off will not destroy the settings.

In this diagram the 'Input action' decision represents any departure from the basic operation which will be addressed next.

Processing Weight Samples and Zero.

The samples buffer will be processed to produce a fixed point number proportional to weight only. – The significance is that no units conversion is done on the raw data to preserve as much precision as possible.

At the most basic level the processing is an arithmetic mean but will likely develop through the use of frequency calculations. – This module will effectively take the samples buffer and output some representation of the weight. Algorithms can be tested and run in parallel to optimise this module. What is most important is that this is not a development show-stopper despite its obvious importance to the product.

Uses function prototype:

```
int Process_Samples (int * buffer, int sampleCount);
```

The 'TARE' function will take the current processed value and set it as the relative zero point. 'TARE' produces a unit-independent weight value 'currZero'. Effectively:

```
currZero = Process_Samples(sampleBuffer, sampleCount);
```

Meaningful usage may be:

```
currWeight = Process_Samples(sampleBuffer, sampleCount) – currZero;
```

Divide by Unit Divisor

Once a raw weight is determined, it is divided by a divisor corresponding to the desired unit. The unit divisor and name are stored as elements in a 'unit' struct. The reasoning is that as a possible functional addition, the user will be able to define several favourite units, all maintained within EEPROM.

This diagram shows the logic behind unit selection and the count function which is invoked whenever the user produces an input for grams, ounces or count.

'currUnit' points to the active unit struct.

Display Weight (or Count) and Process Input Buffer

The display method is defined within 'mode' structs UserLocal, UserRemote and Factory. – Each of these 'mode' structs will contain certain boolean values for the operation of the system such as whether it is local or remote and whether factory functions are allowed or not.

The pointer currMode will point to the active user mode.

Uses function prototype:

```
char Display (struct mode* currMode, struct unit* currUnit, int currWeight);
```

Important to note that while Serial and LCD are continuous display methods, the speech generator is a discrete display method only. This means that the speech generator should only be called once the weight reading has stabilised.

Input methods, either local and remote, are always ready and will fill an 'input buffer', which is progressively emptied in each basic operation iteration.

Inputs will be handled by hardware as either external interrupts for keypad or RX interrupts for RS232.

In the majority of cases, the input will require the program to break from basic operation and perform a settings change, such as changing the value of currUnit or currMode. - An idea we are playing with is switching user modes on detection of alternate input method. That is, when any character is detected via RS232 the machine goes into Remote Mode and when a keypad button is registered, the machine goes into Local Mode.

Factory Functions

Of the four Factory mode only functions, show_statistics and show_weight_readings and set_samples_per_measurement require no imagination.

Only CALIBRATE is interesting in the Factory functions.

Some interpretations we have come up with:

1. Redefine 'gram' and 'ounce' values. – In production this is an important step. Material properties such as modulus of elasticity or non-homogeneity may effect the readings.
2. Redefine the upper and lower reference voltages for the ADC module. – Again in production, component tolerance and operating conditions could change the electrical properties of the amplifier circuit used in the scales.
3. Plot significant points as used by the 'Process_Samples' function. – Perhaps the obvious one but the raw voltage will likely not be perfectly linear to the weight. This may need a bit of mathematics to approximate efficiently.